SMPBS: Web Server for Computing Biomolecular Electrostatics Using Finite Element Solvers of Size Modified Poisson-Boltzmann Equation

Yang Xie,[a] Jinyong Ying,[b] and Dexuan Xie*[b]

SMPBS (Size Modified Poisson-Boltzmann Solvers) is a web server for computing biomolecular electrostatics using finite element solvers of the size modified Poisson-Boltzmann equation (SMPBE). SMPBE not only reflects ionic size effects but also includes the classic Poisson-Boltzmann equation (PBE) as a special case. Thus, its web server is expected to have a broader range of applications than a PBE web server. SMPBS is designed with a dynamic, mobile-friendly user interface, and features easily accessible help text, asynchronous data submission, and an interactive, hardware-accelerated molecular visualization viewer based on the 3Dmol.js library. In particular, the viewer allows computed electrostatics to be directly mapped onto an irregular triangular mesh of a molecular surface. Due to this functionality and the fast SMPBE finite element solvers, the web server is very efficient in the calculation and visualization of electrostatics. In addition, SMPBE is reconstructed using a new objective electrostatic free energy, clearly showing that the electrostatics and ionic concentrations predicted by SMPBE are optimal in the sense of minimizing the objective electrostatic free energy. SMPBS is available at the URL: *smpbs.math.uwm.edu* © 2017 Wiley Periodicals, Inc.

DOI: 10.1002/jcc.24703

Introduction

Calculation of biomolecular electrostatics in an ionic solvent is one fundamental task in computational chemistry. As a commonly used implicit solvent model,[1,2] the Poisson-Boltzmann equation (PBE) has been widely applied to the prediction of electrostatic solvation and binding free energies due to the popularity of software packages such as APBS,[3] DelPhi,[4] PBSA,[5,6] UHBD,[7] and PBEQ[8,9] as well as their web servers. However, in PBE, ions are distinguished only by charge. Thus, for the applications that need to distinguish ions by size (e.g., Na^+ and K^+), PBE may work improperly. To remedy this weakness of PBE, research was done in the last decades to develop dielectric continuum models that consider ionic size effects.[10–20] One of them is a simple size modified PBE (SMPBE), in which each ion is assumed to occupy the same volume of a cube with side length Λ.[11] The solution existence and uniqueness of SMPBE was proved in Refs. [12,21], where the ionic concentrations determined by SMPBE were also shown to be optimal in the sense of minimizing an electrostatic free energy function.

SMPBE is an extension of PBE in terms of the ionic size parameter Λ. It can have a broader range of applications than PBE as it not only reflects ionic size effects but also includes PBE as a special case (with $\Lambda = 0$). Furthermore, the exponential nonlinearity of PBE can be significantly restrained with a proper selection of Λ, making SMPBE easier to solve numerically than PBE in most cases. Hence, SMPBE can be a good substitute for PBE in the calculation of electrostatics and ionic concentrations.

Recently, we developed two efficient SMPBE solvers[21,22] as the extensions of our fast PBE finite element solver[23] and our PBE finite element and finite difference hybrid solver,[24] respectively. Our PBE and SMPBE solvers were constructed using advanced numerical techniques such as finite element, solution decomposition, domain decomposition, and multigrid preconditioning. They have been shown to be efficient and stable numerically, and to be able to generate numerical solutions in a high order of accuracy, especially in the calculation of electric force fields. However, because their software packages were written in C++, C, Fortran, and Python, and rely on the state-of-the-art finite element library DOLFIN from the FEniCS project,[25] they are difficult to install and implement by users who lack sufficient knowledge in programing and scientific computing. This motivates us to develop web servers so that our PBE and SMPBE solvers can be used remotely by any person with an internet connection.

We began web server development about one year ago, releasing the PBE web server SDPBS,[26] which has been moved to a new URL: *sdpbs.math.uwm.edu*. Here, we developed a visualization tool based on JSmol (or Jmol http://jmol.sourceforge.net/)—one of most widely used tools to support three dimensional visualization of molecular structures of biomolecules and chemical compounds. However, JSmol works

[a] *Y. Xie*
Department of Computer Science, University of Wisconsin-Milwaukee, Milwaukee, Wisconsin 53201
[b] *J. Ying, D. Xie*
Department of Mathematical Sciences, University of Wisconsin-Milwaukee, Milwaukee, Wisconsin 53201
E-mail address: dxie@uwm.edu

Contract grant sponsor: National Science Foundation (USA); Contract grant number: DMS-1226259

only for a uniform box mesh. To use JSmol for visualizing our finite element PBE solution, we had to interpolate our electrostatic potential results from an irregular tetrahedral mesh into a uniform cube mesh, and did interpolation again to map the electrostatic potential onto a molecular surface mesh generated from JSmol. Such a visualization procedure was very expensive in CPU time, seriously decaying the performance of SDPBS.

Our development of the SMPBE web server, which we call size modified Poisson Boltzmann solvers (SMPBS), represents further refinement of our work on SDPBS. In addition to the implementation of a new core model, we switched our molecular visualization viewer from JSmol to a new hardware-accelerated graphics library, 3Dmol.js,[27] and revamped the user interface to be less cluttered. As 3Dmol.js facilitates an irregular mesh representation, we wrote custom Javascript routines to yield a new visualization tool with which we can view our electrostatic results and associated molecular surface in their original form. The resulting tool renders user interactions with the model, such as rotations, more smoothly. In general, the revamped workflow of the web server, including the generation and visualization of the molecular surface, finite element mesh, and electrostatics, takes place in around a minute only. For example, the total CPU time for a protein with 11,439 atoms, solved using an irregular tetrahedral mesh with 32,553 vertices, was about 40 sec. This was performed on a single core of a Mac Pro workstation with a 3.7 GHz Quad-Core Intel Xeon E5 processor and 64 GB of RAM.

There are several ways to construct SMPBE from the literature.[11,12] In this article, we present a new construction of SMPBE using a new objective function of electrostatic free energies. This new work further validates the optimality of the electrostatics and ionic concentrations determined by PBE and SMPBE. It will be helpful for users to study PBE and SMPBE.

The remaining part of the article is arranged as follows. In SMPBE Definition and Applications section, we introduce the SMPBE model and its two important applications from the web server. In Web Server Design and Features section, we present the web server design and features. In Conclusions section, we present our conclusions. Finally, a new construction of SMPBE is presented in the Appendix for clarity.

SMPBE Definition and Applications

In this section, we first introduce the size modified Poisson-Boltzmann equation (SMPBE) and its two numerical solvers used by the web server. We then describe two applications of the web server. Numerical results are also reported to demonstrate the performance of the web server.

Size modified Poisson-Boltzmann equation

Let u be an electrostatic potential function, and Ω be a bounded domain split in the form

$$\Omega = D_p \cup D_s \cup \Gamma,$$

where D_p is a protein region hosting a protein (or other biomolecules) with n_p atoms, D_s is a solvent region containing a

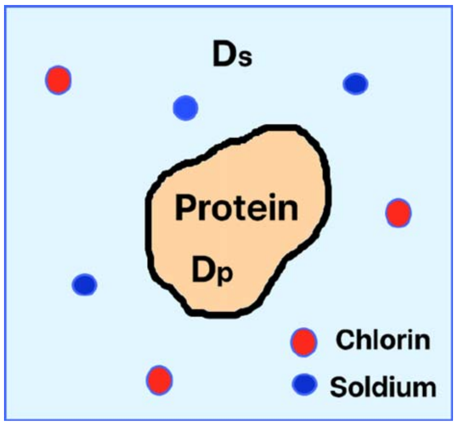


Figure 1. An illustration of a protein region D_p surrounded by a solvent region D_s containing sodium ions (Na^+) and chloride ions (Cl^-). [Color figure can be viewed at wileyonlinelibrary.com]

symmetric 1:1 ionic solvent (such as the salt solution with sodium ions (Na^+) and chloride ions (Cl^-)), and Γ denotes the interface between D_p and D_s. Here, Ω, D_p, and D_s are open domains, D_p is surrounded by D_s, and Γ is usually set as a molecular surface of the protein. See Figure 1 for an illustration.

Based on the implicit solvent approach, both D_p and D_s are treated as continuum media with dielectric constants ε_p and ε_s, respectively. The electrostatic potential u can then be calculated in our web server as a numerical solution of the SMPBE model:

$$\begin{cases} -\varepsilon_p \Delta u(\mathbf{r}) = \dfrac{10^{10} e_c^2}{\varepsilon_0 k_B T} \displaystyle\sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\ -\varepsilon_s \Delta u(\mathbf{r}) + \dfrac{2 I_s N_A e_c^2 \sinh(u)}{10^{17} \varepsilon_0 k_B T [1 + \frac{2}{10^{27}} N_A I_s \Lambda^3 \cosh(u)]} = 0, & \mathbf{r} \in D_s, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ u(\mathbf{s}) = g(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{1}$$

where $\varepsilon_0, e_c, k_B, T, N_A$, and I_s are the permittivity of the vacuum, the electron charge, the Boltzmann constant, the absolute temperature, the Avogadro number, and the ionic strength, respectively, Λ is the ionic size parameter, $\partial\Omega$ denotes the boundary of Ω, $\mathbf{r}_j$ and z_j are the position and charge number of atom j of the protein, $\delta_{\mathbf{r}_j}$ is the Dirac delta distribution at point $\mathbf{r}_j$, $\mathbf{n}(\mathbf{s})$ denotes the unit outward normal vector of D_p, g is a given boundary value function, $\frac{\partial u(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}$ is the directional derivative along direction $\mathbf{n}(\mathbf{s})$, $\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}$ for $\mathbf{r} = (x, y, z)$, $u(\mathbf{s}^\pm) = \lim_{t \to 0^+} u(\mathbf{s} \pm t\mathbf{n}(\mathbf{s}))$, and $\frac{\partial u(\mathbf{s}^\pm)}{\partial \mathbf{n}(\mathbf{s})} = \lim_{t \to 0^+} \frac{\partial u(\mathbf{s} \pm t\mathbf{n}(\mathbf{s}))}{\partial \mathbf{n}(\mathbf{s})}$.

The above SMPBE model has been reformulated in the dimensionless form under the SI (Le Système International d' Unités) units system except that the length unit has been changed from meters to angstroms (Å). Thus, protein configuration data from the Protein Data Bank (PDB) (http://www.rcsb.org/) can be directly used for the calculation.

When the SMPBE solution u is found, the concentrations C_+ and C_- of cations and anions (such as Na^+ and Cl^-) can be calculated by the analytical expressions:

$$\begin{aligned} C_+(\mathbf{r}) &= \frac{I_s e^{-u(\mathbf{r})}}{1+\frac{2}{10^{27}}N_A I_s \Lambda^3 \cosh[u(\mathbf{r})]}, \\ C_-(\mathbf{r}) &= \frac{I_s e^{u(\mathbf{r})}}{1+\frac{2}{10^{27}}N_A I_s \Lambda^3 \cosh[u(\mathbf{r})]} \quad \forall \mathbf{r} \in D_s. \end{aligned} \tag{2}$$

Clearly, with $\Lambda = 0$, SMPBE is reduced to the classic PBE model as follows:

$$\begin{cases} -\varepsilon_p \Delta u(\mathbf{r}) = \dfrac{10^{10} e_c^2}{\varepsilon_0 k_B T} \displaystyle\sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\ -\varepsilon_s \Delta u(\mathbf{r}) + \dfrac{2 I_s N_A e_c^2}{10^{17} \varepsilon_0 k_B T} \sinh(u) = 0, & \mathbf{r} \in D_s, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ u(\mathbf{s}) = g(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{3}$$

and the expressions of (2) are reduced to the ones for PBE:

$$C_+(\mathbf{r}) = I_s e^{-u(\mathbf{r})}, \quad C_-(\mathbf{r}) = I_s e^{u(\mathbf{r})} \quad \forall \mathbf{r} \in D_s. \tag{4}$$

Hence, SMPBE contains PBE as a special case. Conversely, from eqs. (1) and (3) it can be seen that the nonlinearity of SMPBE has been restrained by an nonzero ionic size parameter Λ. Thus, SMPBE not only can reflect ionic size effects but also can be more easily solved numerically than PBE in general. Moreover, as shown in Refs. [21,22], SMPBE can yield more physically reasonable ionic concentrations than PBE.

SMPBE solvers for the web server

In this web server, a finite element solver reported in Refs. [21,23] and a finite element and finite difference hybrid solver reported in Ref. [22] have been adopted to solve SMPBE numerically. To avoid the singularity difficulties caused by the atomic point charges from the protein, these two solvers have been developed using the following solution decomposition:

$$u = G + \Psi + \tilde{\Phi}, \tag{5}$$

where G is given by the analytical expression

$$G(\mathbf{r}) = \frac{10^{10} e_c^2}{4\pi \varepsilon_p \varepsilon_0 k_B T} \sum_{j=1}^{n_p} \frac{z_j}{|\mathbf{r} - \mathbf{r}_j|}, \tag{6}$$

Ψ is a solution of the linear interface boundary value problem

$$\begin{cases} \Delta \Psi(\mathbf{r}) = 0, & \mathbf{r} \in D_p \cup D_s, \\ \Psi(\mathbf{s}^+) = \Psi(\mathbf{s}^-), & \mathbf{s} \in \Gamma, \\ \varepsilon_s \dfrac{\partial \Psi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial \Psi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} + (\varepsilon_p - \varepsilon_s) \dfrac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \Psi(\mathbf{s}) = g(\mathbf{s}) - G(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{7}$$

and $\tilde{\Phi}$ is a solution of the nonlinear interface boundary value problem

$$\begin{cases} \Delta \tilde{\Phi}(\mathbf{r}) = 0, & \mathbf{r} \in D_p, \\ -\varepsilon_s \Delta \tilde{\Phi}(\mathbf{r}) + \dfrac{2 I_s N_A e_c^2 \sinh(G + \Psi + \tilde{\Phi})}{10^{17} \varepsilon_0 k_B T [1 + \frac{2}{10^{27}} N_A I_s \Lambda^3 \cosh(G + \Psi + \tilde{\Phi})]} = 0, & \mathbf{r} \in D_s, \\ \tilde{\Phi}(\mathbf{s}^+) = \tilde{\Phi}(\mathbf{s}^-), \varepsilon_s \dfrac{\partial \tilde{\Phi}(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial \tilde{\Phi}(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \tilde{\Phi}(\mathbf{s}) = 0, & \mathbf{s} \in \partial\Omega. \end{cases} \tag{8}$$

Here $\frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} = \nabla G \cdot \mathbf{n}$ with ∇G being given by

$$\nabla G(\mathbf{r}) = -\frac{10^{10} e_c^2}{4\pi \varepsilon_p \varepsilon_0 k_B T} \sum_{j=1}^{n_p} z_j \frac{\mathbf{r} - \mathbf{r}_j}{|\mathbf{r} - \mathbf{r}_j|^3}. \tag{9}$$

Because G contains all the singularity points of u, both Ψ and $\tilde{\Phi}$ become well defined without any singularity so that they can be much easier to compute numerically than u. The five basic steps of our two SMPBE solvers are given as follows:

- **Step 1**: Calculate both G and ∇G via the analytical formulas (6) and (9).
- **Step 2**: Generate an interface fitted tetrahedral mesh of domain Ω via our mesh generator.
- **Step 3**: Solve the linear interface problem (7) for Ψ via our finite element linear solver.
- **Step 4**: Solve the nonlinear interface problem (8) for $\tilde{\Phi}$ via our finite element nonlinear solver.
- **Step 5**: Construct the solution u of SMPBE via the solution decomposition (5).

The above solution u and its three components G, Ψ, and $\tilde{\Phi}$ are saved to a data file for downloading by default. It is a dimensionless electrostatic potential, and can be converted to the electrostatic potential Φ in volts via the transform

$$\Phi(\mathbf{r}) = \frac{k_B T}{e_c} u(\mathbf{r}), \quad \mathbf{r} \in \Omega.$$

The software packages of these two solvers were written in C++, C, Fortran, and Python based on the state-of-the-art finite element library from the FEniCS project.[25] Our mesh generator, called GAMer II, is an extension of a molecular surface and volumetric mesh generation program package, GAMer, reported in Ref. [31]. It can generate molecular surface-fitted tetrahedral meshes for rectangular and spherical

Table 1. A performance comparison between our SMPBE and PBE finite element solvers for four proteins using the default values of the web server SMPBS.

PDB ID (Atom number)	N_h	CPU time in seconds (percentage)					Solvation energy (kcal/mol)
		Mesh	G	Ψ	$\tilde{\Phi}$	Total	
		Case of SMPBE using Λ=3.11					
4PTI (892)	71427	15.47 (46.61%)	1.21	1.32	15.18	33.18	−323.24
2JK4 (4393)	65655	19.46 (48.31%)	5.47	1.28	14.06	40.29	−623.28
2AQ5 (6024)	62099	19.729 (47.14%)	7.03	1.01	14.07	41.85	−866.27
1C4K (11439)	32553	22.02 (58.68%)	6.98	0.42	8.10	37.53	−3148.47
		Case of PBE (i.e., SMPBE using Λ = 0)					
4PTI (892)	71427	15.50 (42.62%)	1.21	1.23	18.42	36.36	−323.62
2JK4 (4393)	65655	19.42 (39.56%)	5.38	1.28	23.02	49.10	−626.32
2AQ5 (6024)	62099	19.511 (38.21%)	6.96	1.06	23.53	51.06	−870.95
1C4K (11439)	32553	22.84 (45.58%)	6.90	0.53	19.84	50.11	−3191.94

Here, N_h is the number of mesh points of a tetrahedral mesh.

domains and three molecular interfaces — the Gaussian surface, the solvent-excluded surface (SES), and the solvent-accessible surface (SAS).[31,32] It can also generate a mesh mixing a uniform Cartesian mesh with an unstructured finite element mesh for the hybrid solver.[24]

The Gaussian surface is set as the default choice in the web server SMPBS as it has been modified in Ref. [31] specifically for generating a mesh for the finite element solver. In contrast, SES and SAS were adopted to GAMer II without any change yet. Their molecular surface triangular meshes may contain improper triangles (e.g., with an angle close to 180 degree or a vertex lying on an edge of another triangle), which may cause GAMer II to crash. However, when a biomolecule has a highly complex shape, the web server may fail to generate a mesh occasionally even with the Gaussian surface. In this case, the user is advised to adjust the default values of mesh parameters and try again. A brief description on the usages of our mesh parameters has been given in Ref. [26]. Further guidance and usage details can be found in Ref. [31] for the Gaussian surface and Ref. [32] for SES and SAS.

The software packages of our two SMPBE solvers have been reorganized with many modifications to match the needs of the web server. A protein PQR file is required by our solvers, which is a modified PDB file containing the hydrogen atoms (or some other atoms), the atomic charge numbers, and atomic radii that are not included in a PDB file. It can be generated from the PDB2PQR web server (http://nbcr-222.ucsd.edu/pdb2pqr_2.1.1/).

To demonstrate the performance of our solvers, we made simple tests using the default values so that these tests can be repeated by any user. In these tests, the PQR files were generated from the PDB2PQR web server by entering PDB IDs and selecting the AMBER force field. They were then uploaded to our web server to run jobs without adjusting any parameter. These jobs were run on our Mac Pro Workstation with 3.7 GHz Quad-Core Intel Xeon E5 processor and 64 GB memory. For comparison, we repeated the tests, changing the values of the ionic size parameter Λ from 3.11 to 0, which gave the performance of the classic PBE model. The PDB IDs and the CPU time data were reported in Table 1. The convergence behaviors of our nonlinear iterative method for solving the nonlinear interface problem (8) were displayed in Figure 2. From the table and figure, it can be seen that our finite element solver was very efficient for both SMPBE and PBE. They also confirm that SMPBE can be solved more efficiently by our solver than PBE because SMPBE has weaker nonlinearity than PBE.

We noticed that the mesh generator cost about a half of the total CPU time. In the web server, the mesh files have been set to save by default to reuse them in the future calculation.

Calculation of solvation and binding free energies

The web server SMPBS provides users with services for computing electrostatic solvation and binding free energies, which are two important applications of SMPBE.

Electrostatic solvation energy. In kilocalories per mole (kcal/mol), the electrostatic solvation free energy E is defined by

$$E=\frac{N_A}{4184}\frac{k_BT}{2e_c}\int_\Omega \rho_p(\mathbf{r})(u_{sol}-u_{ref})(\mathbf{r})d\mathbf{r}, \tag{10}$$

where u_{ref} denotes the electrostatic potentials in a reference state, and ρ_p denotes a fixed charge density function. In SMPBS, $\rho_p=e_c\sum_{j=1}^{n_p} z_j\delta(\mathbf{r}-\mathbf{r}_j)$ and $u_{ref}=G$. With the solution decomposition (5), we can reformulate (10) as a new formula without involving any singularity:

$$E=\frac{N_A}{4184}\cdot\frac{k_BT}{2}\sum_{j=1}^{n_p} z_j\left[\Psi(\mathbf{r}_j)+\tilde{\Phi}(\mathbf{r}_j)\right]. \tag{11}$$

To demonstrate the numerical stability of our two SMPBE solvers, we did tests for 95 proteins in the calculation of solvation energy. Here, the 95 proteins were downloaded from the website http://rayl0.bio.uci.edu/rayl/, and were tested on six sets of meshes with the average numbers of mesh points in the range of 20,997 to 237,441. To present the test results in one plot, we calculated the average mesh point numbers and average solvation energies over the 95 proteins. As the analytical value of E is unknown, we took the value of E from the finest mesh,

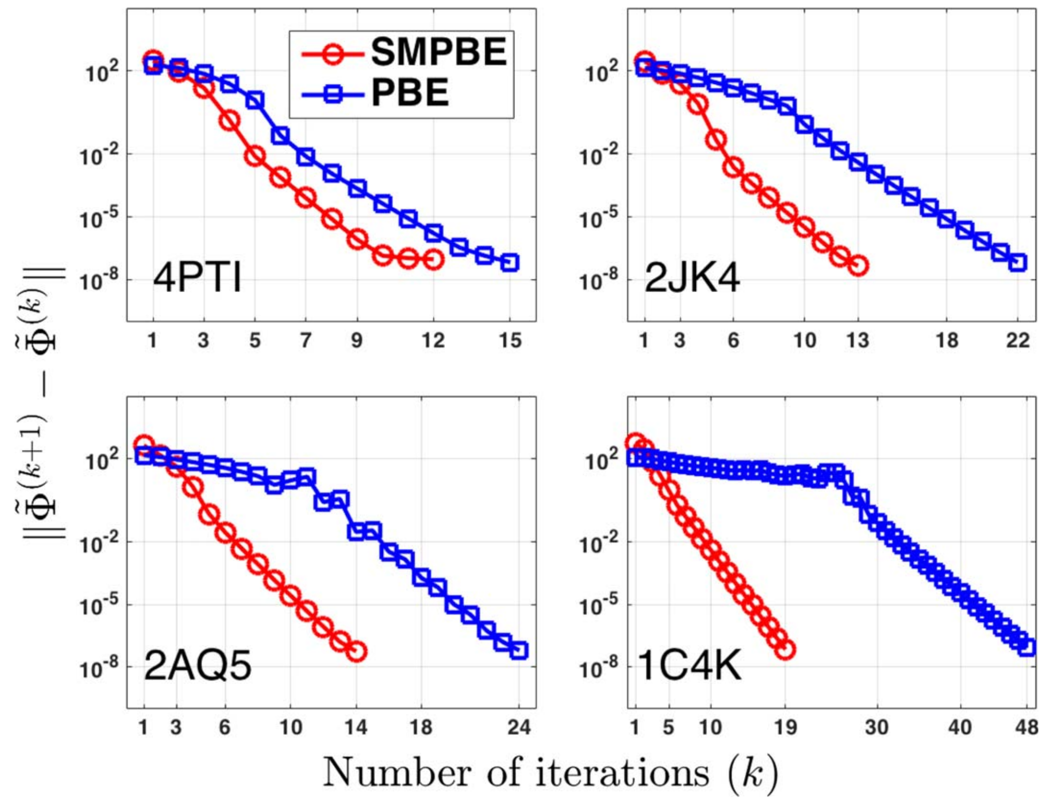


Figure 2. A comparison of the convergence of the SMPBE finite element solver with that of the PBE finite element solver (i.e., the case of $\Lambda = 0$) in the numerical solution of the nonlinear interface problem (8) for $\tilde{\Phi}$. Here, $\tilde{\Phi}^{(k)}$ denotes the kth iterate, and $||\cdot||$ is the L_2 norm. [Color figure can be viewed at wileyonlinelibrary.com]

which has 237,441 mesh points, as the reference to calculate the relative errors for the other five values of E. The test results are reported in Figure 3.

Figure 3 shows that the relative error decreases monotonically from 0.015 to 0.001 as the average number of mesh points increases from 20,997 to 136,435, implying the numerical stability of our two solvers in the calculation of solvation energy.

It should be noted that the number of mesh points can only be indirectly adjusted, via related parameters such as "Coarsening rate," "Marching cube side length of domain," "Minimum radius-edge ratio," "Maximum volume," and "Central box partitions" (which is for the hybrid solver only). Guidance on how these parameters affect the number of mesh points are available through tooltips presented in the web server user interface. In the tests as reported in Figure 3, we increased the value of "Marching cube side length of domain" from 1.5 to 1.99 for all the meshes. We then set "Minimum radius-edge ratio" as 1.5, 1.2, 1.2, 1.2, 1.1, 1.1, and "Maximum volume" as 1000, 100, 10, 5, 1, 0.5, respectively, to yield the six sets of meshes.

Electrostatic binding free eneregy. Calculation of a binding free energy, E_b, is often required to study the salt dependance of two molecules A and B binding as a complex C.[33–35] As another important application of SMPBS, we calculate E_b using the following formula:

$$E_b(I_s)=E(C,I_s)-E(A,I_s)-E(B,I_s), \tag{12}$$

where $E(X, I_s)$ denotes the electrostatic solvation free energy of molecule X from the vaccum state to the salt solution state with the ionic strength $I_s \in (0, 1)$. In particular, we calculate $E(X, I_s)$ using formula (11). Because Ψ is independent of I_s, $E(X, I_s)$ is further simplified as follows:

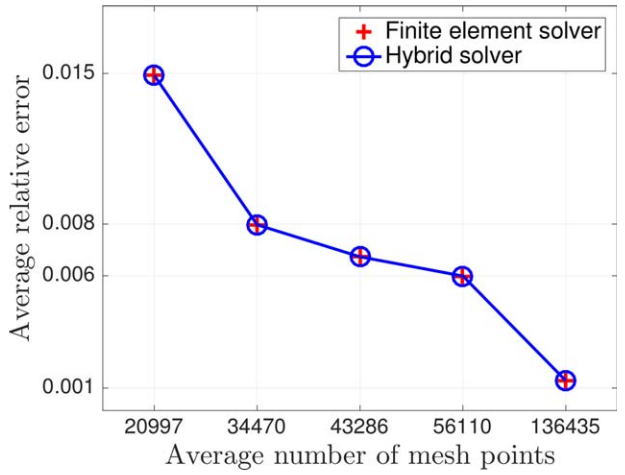


Figure 3. Relative errors of the average solvation energies calculated by SMPBE over 95 proteins using six sets of meshes with the reference value calculated using the meshes with the average mesh number being 237,441. [Color figure can be viewed at wileyonlinelibrary.com]

$$E(X,I_s)=\frac{N_A}{4184}\cdot\frac{k_BT}{2}\sum_{j=1}^{n_p}z_j\tilde{\Phi}_{X,I_s}(\mathbf{r}_j), \tag{13}$$

where $\tilde{\Phi}_{X,I_s}$ denotes a solution of the nonlinear problem (8) for molecule X in the salt solution with ionic strength I_s. The above formula is particularly efficient as it only needs to calculate $\tilde{\Phi}_{X,I_s}$ after G and Ψ are calculated for the first value of I_s.

Via the variable change $\xi=\log_{10}I_s$, the binding free energy E_b becomes a linear function of ξ with the slope m on an interval $[a_1,a_2]$ of ξ[36]:

$$E_b=m\xi+b, \quad a_1\le\xi\le a_2. \tag{14}$$

In the web server SMPBS, the slope m is calculated in three steps:

1. Set $\xi_j=a_1+j\tau$ for $j=0,1,2,\ldots,l$, where $\tau>0$, and l is the largest number satisfying $a_1+l\tau\le a_2$.
2. Set $I_{s,j}=10^{\xi_j}$, and calculate binding energy $E_b(I_{s,j})$ for $j=0,1,2,\ldots,l$.
3. Calculate m as the slope of a best fitted line determined by the least square method:

$$m=\frac{\sum_{j=1}^{l}(\xi_j-\bar{\xi})[E_b(I_{s,j})-\overline{E_b}]}{\sum_{j=1}^{l}(\xi_j-\bar{\xi})^2}$$
$$\text{with } \bar{\xi}=\frac{1}{l}\sum_{j=1}^{l}\xi_j \quad\text{and}\quad \overline{E_b}=\frac{1}{l}\sum_{j=1}^{l}E_b(I_{s,j}). \tag{15}$$

Another variable change, $\xi=\ln I_s$, is also used to reformulate E_b as a linear function with slope m_s. It is easy to get that $m_s=m\log_{10}e$. Here, $\log_{10}e\approx 0.4343$.

On our web server's user interface, a default value of the ionic size parameter Λ has been set as 3.11, which we obtained as shown here. Let Λ be the side length of a cube that hosts one water molecule. At temperature 298.15 Kelvin, it is known that the density of liquid water is 997,047.9 grams per cubic meter, and the molar mass of water is 18.01528 grams per mole. Thus, Λ can be estimated as follows:

$$\Lambda=\sqrt[3]{\frac{18.01528\times10^{30}}{997047.9N_A}}=3.10735\ldots\approx 3.11\text{ Å},$$

where we have assumed that there are N_A water molecules per mole, and used 1 meter $=10^{10}$ Å.

To demonstrate that the default value 3.11 of Λ is a reasonable choice, we did tests on the calculation of binding free energy for a DNA-drug complex (PDB ID 1D86) using SMPBE with $\Lambda=0,0.5,1,1.5,2,2.5,3.11,3.5,4$. In these tests, we used $\tau=0.2$ and $l=11$ for $a_1=-3$ and $a_2=-1$, which gave $0.05\le I_s\le 0.37$. The three meshes had 259,903, 261,512, and 233,983 mesh points for the complex and the two components, respectively. The experimental slope $m_e=0.8946$ kcal/mol was produced from the scaled experimental slope $m_s=-1.51$[37] by the formula $m_e=-m_sN_Ak_BT/4184$. We then calculated the relative error using m_e as the reference.

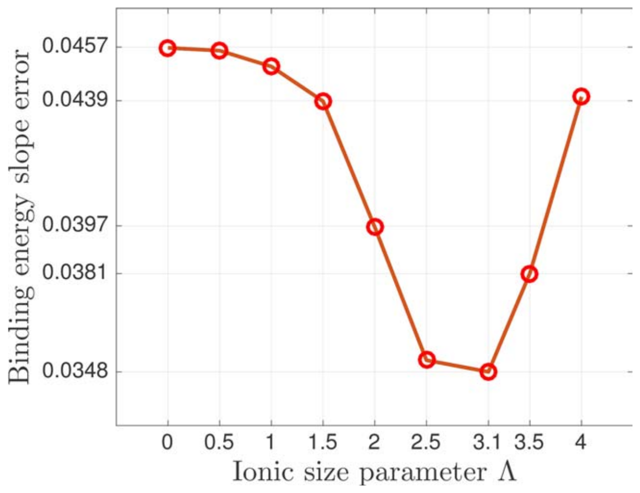


Figure 4. Relative error as a function of Λ for the DNA-drug complex (PDB ID 1D86).[34] Here, the relative error is given by $|m-m_e|/m_e$ with m being a slope predicted by SMPBE and $m_e=0.894$ (an experimental slope[34]). [Color figure can be viewed at wileyonlinelibrary.com]

The test results are reported in Figure 4, from which it can be seen that the relative error of the slope determined by PBE (the case with $\Lambda=0$) can be significantly reduced by SMPBE with $\Lambda=3.11$, which supports its selection as the default value for Λ on our web server's user interface.

Web Server Design and Features

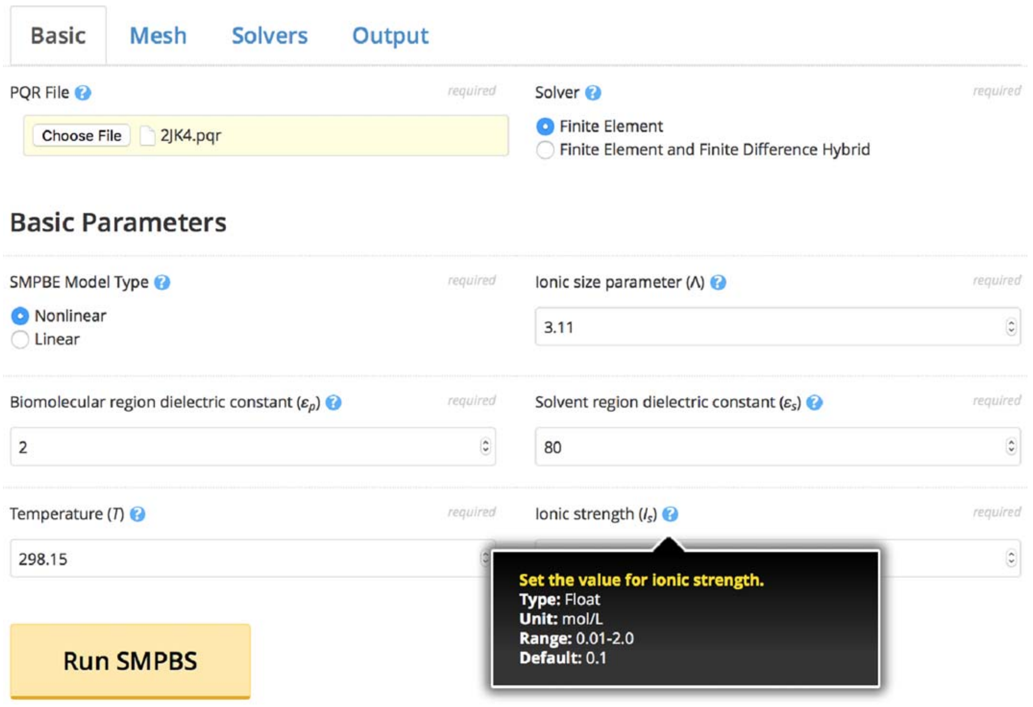
Our web server consists of four web pages titled *Home, Features, Solvation Energy, and Binding Energy* according to their major functions.

Home and features

These two web pages provide users with basic information on SMPBE and its web server SMPBS. On the web page titled *Home*, a short introduction is given on the definition of SMPBE, the two SMPBE solvers, and the basic usage of the web server as well as other information such as references and credits/acknowledgements to the organizations that supported our research and the people who made contributions to the web server development. All the parameters of SMPBE are listed in three tables with their definitions, units, and values. As SMPBE is not as well known as PBE, a common question that may be asked is why SMPBE should be considered for computing ionic concentrations. To address that question, the web page titled Features presents one simple numerical example that was studied in our article[22]—a dipole model with two opposite central charges. The numerical results from this dipole model simulation demonstrate that SMPBE can yield more physically reasonable ionic concentrations than PBE.

Solvation energy and binding energy

These web pages contain the major functionalities of the web server SMPBS. Here, users can submit a job to compute either



PQR File (required): 2JK4.pqr

Solver (required): [✓ Finite Element] Finite Element and Finite Difference Hybrid

Basic Parameters

SMPBE Model Type (required): [✓ Nonlinear] Linear

Ionic size parameter (Λ) (required): 3.11

Biomolecular region dielectric constant (ε_p) (required): 2

Solvent region dielectric constant (ε_s) (required): 80

Temperature (*T*) (required): 298.15

Ionic strength (I_s) (required)

Set the value for ionic strength.
Type: Float
Unit: mol/L
Range: 0.01-2.0
Default: 0.1

Figure 5. A snapshot of the Parameters tab with basic parameters exposed. Here, a PQR file, 2JK4.pqr, is selected for upload, and a tooltip is displayed for the parameter I_s. [Color figure can be viewed at wileyonlinelibrary.com]

the electrostatic solvation energy of a biomolecule or the binding free energy of a biomolecule complex, and view the results returned from the server.

To illustrate the general layout of related web pages, we will focus on the Solvation Energy page as an example. The main content of the page is grouped into three categories, which are named as Parameters, Visualization & Results, and Output Log, respectively. The content of these categories are hidden or displayed when the user clicks on one of the three corresponding tabs on the page.

Parameters. In this tab, users can upload a PQR file, choose a SMPBE solver, adjust additional parameters, and submit a job. We exposed most of the parameters for our solvers and mesh generator as adjustable fields on our web user interface. Thus, users have a high degree of control over the default values of parameters that we set for their job submission. Because of this feature, the web server can be a valuable education tool for users to study the properties of SMPBE and to explore the performance of our finite element and hybrid solvers.

Most of the user-adjustable parameters are grouped into four categories, named Basic, Mesh, Solvers, and Output, and are hidden within four corresponding tabs on the page. The PQR file and the solver type fields are considered crucial and are thus displayed persistently across each tab. A tooltip explaining the definition and usage of a parameter can be displayed when hovering the mouse pointer over a question mark beside each parameter label.

Figure 5 contains a snapshot of the Parameters tab when its Basic sub-tab is activated, displaying parameters such as SMPBE model type, the two dielectric constants ε_p and ε_s, temperature *T*, and ionic strength I_s.

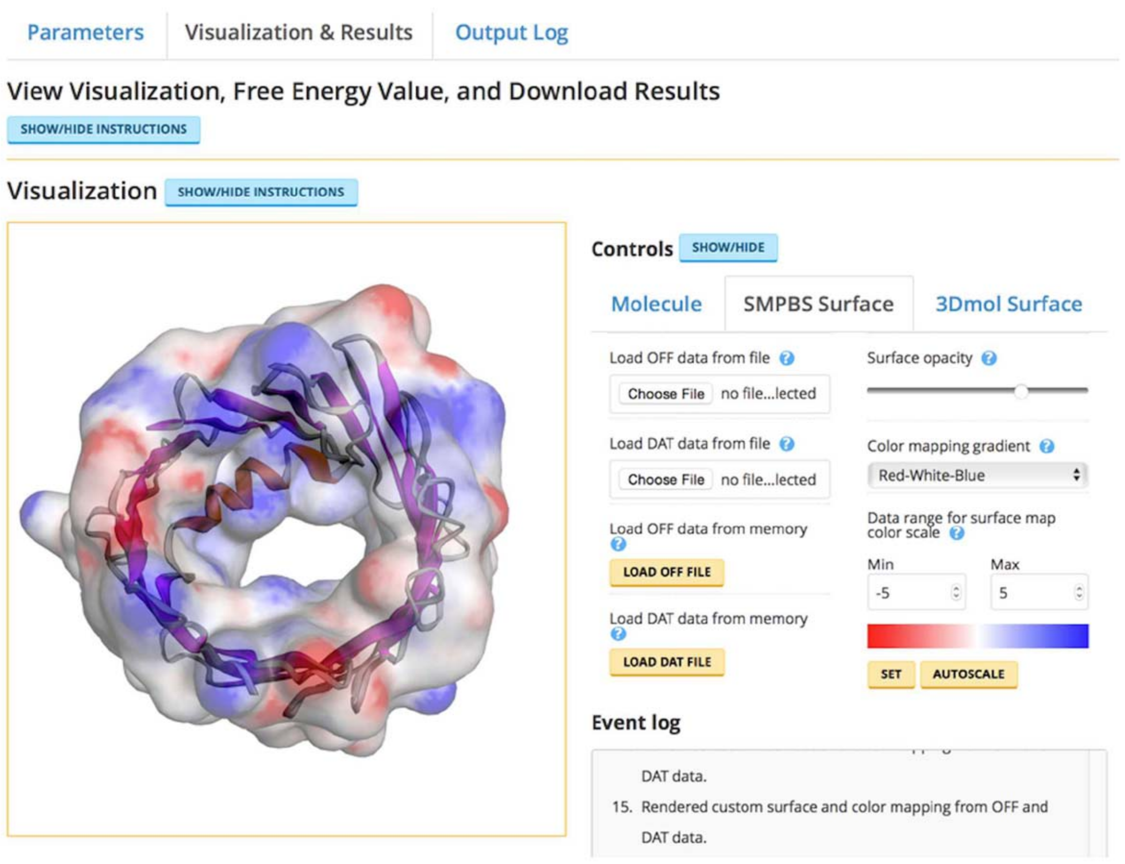
Visualization and Results. In this tab, users can view the results of the job submission. The results are presented as a visualization of the molecular structure and surface of the submitted biomolecule, a calculated solvation energy value, and a downloadable zip file that contains various mesh and data files. Our visualization tool, powered by 3Dmol.js, is accompanied by a tabbed control interface. Users can interact with the tool to view the molecular structure in various styles, view the color mapping of computed electrostatic potential values on a molecular surface generated by our mesh generator GAMer II, and view additional molecular surfaces generated by 3Dmol.js libraries.

Figure 6 is a snapshot of the Visualization and Results tab, which shows a color mapping of electrostatic potential values on a molecular surface of the protein with PDB ID 2JK4.

Our web server also returns a number of files that users can download for further investigation. These include: (1) a copy of the PQR file submitted by the user, (2) a .ini file that contains the solver parameters specified by the user, (3) a .off file that contains surface mesh data, (4) a .dat file that contains a table of electrostatic potential results and their component values, and (5) a .xml file that contains volume tetrahedral mesh data.

In addition, users can enable (via the output parameters tab) the output of .pvd files containing volume tetrahedral mesh data, surface mesh data, protein domain and solvent domain partition information, and electrostatic potential values, as well as .dx files containing the electrostatic potential values at the vertices of a volume cube mesh.

We encourage users to explore our results using external programs. For example, results in our .pvd files can be



Load OFF data from file: Choose File no file...lected

Load DAT data from file: Choose File no file...lected

Surface opacity

Color mapping gradient: Red-White-Blue

Data range for surface map color scale: Min -5 Max 5

```
    DAT data.
15. Rendered custom surface and color mapping from OFF and
    DAT data.
```

Figure 6. A snapshot of a part of the Visualization & Results tab. Here, a color mapping of the electrostatic potential (a solution of SMPBE) on a molecular surface of a human voltage-dependent anion channel (PDB ID: 2JK4) is displayed, and the control tab "SMPBS Surface" has been selected to display various interactive control elements. [Color figure can be viewed at wileyonlinelibrary.com]

analyzed using ParaView (http://www.paraview.org), an open-source, multi-platform data analysis and visualization application, to view isosurfaces and cross-sections. Furthermore, the volume mesh in our .xml files can be used in FEniCS,[25] a free software for automated solution of differential equations, to calculate other aspects of the molecule.

Output Log. In this tab, users can view the raw output of our SMPBE solver, including calculation and mesh generation details, and convergence and performance data.

Server architecture and back-end design

Our web server design follows a standard client-server architecture, in which a client is a computer where the user submits requests for data, and a server is a computer that replies with the requested data. On the client, a user uses a web browser to connect to our website. Using the user interface on the Solvation Energy and Binding Energy pages, a user can upload biomolecules as PQR files, select a SMPBE solver, and adjust various parameters of our mesh generator and SMPBE for the job he/she wishes to submit. This work flow is illustrated in Figure 7.

To ensure the network security, our server (back-end) is executed in a two tier pattern. The first tier is the LAMP (Linux/Apache/MySQL/PHP) server hosted by the Letters and Science (L&S) IT office at the University of Wisconsin-Milwaukee (UWM). This server uses the PHP programming language to prepare and present our website as HTML to the client. It also accepts the files and parameter configurations submitted from the client's browser and relays the data to our second tier through a socket connection across our campus network. The second tier is the computer that runs our SMPBS software packages and performs the main computational duties. It runs a socket program that accepts incoming socket connections from the first tier, and uses the transmitted files and parameter configurations to call the appropriate SMPBS solver routines. After computation by SMPBS is complete, the computer transmits all result data back to the first tier through the socket connection. The first tier receives this data, parses it into meaningful chunks, encodes the data as a JSON (Javascript Object Notation) object, and sends it to the client's browser. The client's browser decodes the JSON data, and our Javascript scripts parse and display the data as readable text, downloadable files, and an interactive visualization on the web page.

Front-end design

We designed the styling of the web server front-end to be intuitive, mobile friendly, and in compliance with university style standards. We primarily used the UWM L&S IT office's internal web development framework, which is built on

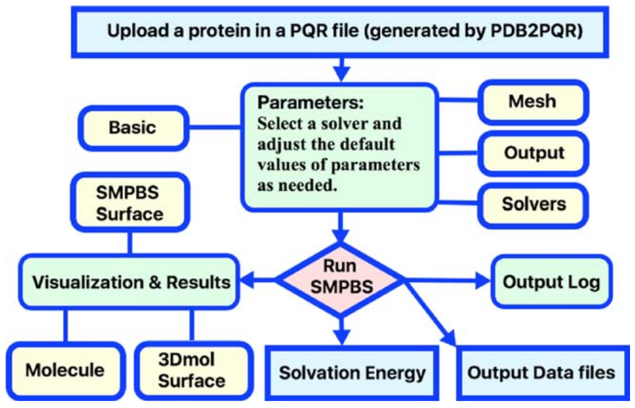

Upload a protein in a PQR file (generated by PDB2PQR)
Basic
Parameters: Select a solver and adjust the default values of parameters as needed.
Mesh
Output
Solvers
SMPBS Surface
Visualization & Results
Run SMPBS
Output Log
Molecule
3Dmol Surface
Solvation Energy
Output Data files

Figure 7. Flow diagram of a job submission on the "Solvation Energy" page. Here, each rounded rectangle represents a tab, and the three major tabs are colored in green. [Color figure can be viewed at wileyonlinelibrary.com]

Bootstrap, a popular front-end framework that enables responsive, mobile-friendly web designs, as well as JQuery, a cross-platform Javascript library that simplifies client-side scripting. We then made extensive use of tabbed elements and toggled instructions to reduce visual clutter and increase user accessibility, and extended the L&S framework to include hoverable tooltips, which provide users with helpful information regarding each adjustable parameter.

Furthermore, we used AJAX (Asynchronous Javascript and XML) calls to facilitate communication between client and server-side scripts. AJAX allows portions of the web page to be updated without requiring a reload, enabling the user to seamlessly interact with the web page during and after submission of a job.

3Dmol.js versus JSmol for visualization

In this subsection, we give a brief overview of JSmol and 3Dmol.js and discuss why we ultimately switched from JSmol to 3Dmol.js to implement molecular visualization in our web server.

As mobile devices are becoming popular, software development is now shifting to web-based or cloud-based platforms, where users can directly visualize 3D graphical models in browsers anywhere, anytime, as long as an internet connection is available. Among several early attempts to this end, Jmol is one of most widely used tools to support 3D visualization of molecular structures of biomolecules and chemical compounds. It has been applied to many web servers, including our SDPBS web server and MIBPB web server. While Jmol is a powerful tool that boasts many sophisticated functions, its implementation as a Java applet has steadily deprecated its use. NPAPI, the application programming interface that enables web browsers to run Java applets, has encountered multiple security issues over the years. As a result, the API is no longer supported by many modern web browsers. In response to Java's deprecation, Jmol was ported, via a language translator, to Javascript, a programming language that is supported by modern web browsers without the need for plug-ins. The resulting tool, JSmol, runs natively on browsers, but its performance is slower than that of Jmol, especially for visualization of large and complex molecules.

In recent years, 3Dmol.js was developed as a more optimized alternative to Jsmol by researchers at the University of Pittsburgh.[27] Like JSmol, 3Dmol.js is written in Javascript and runs without the need for plug-ins on modern web browsers. However, unlike the software-based rendering used in JSmol, 3Dmol.js enables hardware-accelerated rendering using WebGL, which is a standardized Javascript graphical library that taps into the power of modern graphical processing units found in most modern computers.

While 3Dmol lacks some of the advanced functionality of Jmol/JSmol, it still offers the essential functions of rendering, styling, and manipulating molecular data, generating molecular surfaces, and generating custom surfaces from data. Thanks to hardware acceleration, interactions with a 3D scene is much smoother in 3Dmol.js when compared to JSmol.

Irregular mesh visualization enabled by 3Dmol.js

A crucial advantage of using 3Dmol.js is that it allows users to directly visualize the molecular surface mesh and electrostatic potential values generated by our solver. The surface mesh generated from SMPBS is an irregular triangular mesh that more accurately reflects the contours of the molecular surface. However, it is difficult to visualize electrostatics on such an irregular surface mesh. It is known that 3Dmol.js facilitates an irregular surface mesh representation by providing a method to draw a custom shape, given a list of vertex positions, a list of triangles, and a list of colors at each vertex. Taking advantage of this functionality, we wrote Javascript routines to parse vertex positions, triangles, and electrostatic potential values from our output files, and then converted the potential values to RGB (red, green, blue) values and hexadecimal codes based on a color gradient. This data was then used as parameters to draw the custom shape using 3Dmol.js, resulting in a color-mapped surface mesh that is superimposed on the existing molecular model. As far as we are aware, this direct visualization of electrostatics on an irregular triangular surface mesh was never done before. The surface mesh and color mapping are displayed automatically after a job submission completes, requiring no additional input from the user.

Conclusions

In this article, we introduced the web server SMPBS for computing biomolecular electrostatics through solving the SMPBE with our finite element solvers. Because of a new visualization tool powered by 3Dmol.js, the performance of our new web server SMPBS has been sharply increased in comparison to its sister site, the SDPBS web server. For completeness, we reviewed the SMPBE model, the ionic concentration functions, and our new formulas for computing electrostatic solvation and binding free energies, and our solution decomposition-based finite element solvers. Moreover, we reconstructed SMPBE using a new objective energy functional, which clearly illustrates the derivation of SMPBE and verifies the optimality of SMPBE in the prediction of biomolecular electrostatics and ionic concentrations. This is presented in the Appendix for clarity.

Furthermore, numerical results were reported to demonstrate the high performance of our new web server and the numerical stability of our SMPBE solvers. Because users are allowed to adjust the values of most parameters from SMPBE, our mesh generator, and our finite element solvers, and SMPBE includes PBE as a special case, the new web server SMPBS will be a valuable education tool for scientists and students to study both PBE, SMPBE, and their numerical solvers, as well as a powerful tool for researchers to explore various applications.

Acknowledgments

The authors thank Dr. David Koes for his help in integrating 3Dmol.js into our web server. They also thank Dr. Yi Jiang for his help with the SMPBE solver libraries.

Appendix

A new construction of SMPBE

Let the solvent region D_s contain N_1 positively unit charged ions and N_2 negatively unit charged ions. For simplicity, each ion is assumed to have the same volume v. Thus, the volume V of D_s can be expressed by

$$V = V_w + \nu(N_1 + N_2),$$

where V_w denotes the volume occupied by the water solution. Dividing by V on the both sides of the above identity, we obtain the size constraint condition for bulk solvent:

$$\xi^b + \nu(c_1^b + c_2^b) = 1, \tag{1}$$

where $\xi^b = V_w/V$, $c_1^b = N_1/V$, and $c_2^b = N_2/V$, which give the bulk volume fraction and the two bulk concentrations, respectively.

Let c_1 and c_2 be the concentration functions of positively and negatively unit charged ions, respectively. Clearly, the identity (1) can be generalized as the size constraint condition:

$$\xi(\mathbf{r}) + \nu(c_1(\mathbf{r}) + c_2(\mathbf{r})) = 1, \quad \mathbf{r} \in D_s, \tag{2}$$

where $\xi(\mathbf{r})$ is the volume fraction function of the water solution at each point $\mathbf{r}$ of D_s.

With the size constraint condition (2), we construct a new electrostatic energy function F in terms of an electrostatic potential function Φ (in volts) and the two concentration functions c_1 and c_2 (in moles per liter) as follows:

$$F(c_1, c_2; \Phi) = F_{es}(c_1, c_2; \Phi) + F_{id}(c_1, c_2) + F_{ex}(c_1, c_2), \tag{3}$$

where F_{es}, F_{id}, and F_{ex} denote the electrostatic, ideal gas, and excess energies, respectively, as defined by

$$F_{es}(c_1, c_2; \Phi) = \frac{e_c}{2} \int_{D_s} \Phi(\mathbf{r})[c_1(\mathbf{r}) - c_2(\mathbf{r})] d\mathbf{r},$$

$$F_{id}(c_1, c_2) = k_B T \int_{D_s} \left[c_1 \left(\ln \frac{c_1}{I_s} - 1 \right) + c_2 \left(\ln \frac{c_2}{I_s} - 1 \right) \right] d\mathbf{r},$$

and

$$F_{ex}(c_1, c_2) = \frac{k_B T}{\nu} \int_{\Omega} [1 - \nu(c_1 + c_2)][\ln(1 - \nu(c_1 + c_2)) - 1] d\mathbf{r}.$$

Here, the excess energy F_{ex} is produced from the size constraint condition (2).

We now show that the concentrations C_+ and C_- of (2) are optimal in the sense that they are the solution of the Poisson equation constrained minimization problem:

$$\min \{ F(c_1, c_2; \Phi) \mid c_1, c_2 \in L_2(D_s) \}, \tag{4}$$

subject to the Poisson equation

$$-\varepsilon_0 \varepsilon_s \Delta \Phi(\mathbf{r}) = e_c(c_1 - c_2), \qquad \mathbf{r} \in D_s, \quad \Phi(\mathbf{s}) = 0, \quad \mathbf{s} \in \partial D_s, \tag{5}$$

where $L_2(D_s)$ is the L_2 function space with the inner product $\langle u, v \rangle = \int_{D_s} uv d\mathbf{r}$,[28] and D_s has been assumed to be large enough to approximate Φ as zero on the boundary ∂D_s of D_s. Clearly, the Poisson eq. (5) has a unique solution Φ satisfying the variational problem:

$$\text{Find } \Phi \in H_0^1(D_s) \quad \text{such that}$$
$$\varepsilon_0 \varepsilon_s \int_{D_s} \nabla \Phi \nabla v d\mathbf{r} = \int_{D_s} e_c(c_1 - c_2) v d\mathbf{r} \quad \forall v \in H_0^1(D_s),$$

where $H_0^1(D_s)$ is a Sobolev function space as defined in Ref. [28]. Similar to what is done in Ref. [29], we define a continuous self-adjoint positive linear operator L from $H_0^1(D_s)$ to $L_2(D_s)$ by

$$\int_{D_s} v L(u) d\mathbf{r} = \varepsilon_0 \varepsilon_s \int_{D_s} \nabla u \nabla v d\mathbf{r} \quad \forall u, v \in H_0^1(D_s),$$

such that the solution Φ can be expressed as a function of c_1 and c_2 in the operator form

$$\Phi = \Phi(c_1, c_2) \quad \text{with}\, \Phi(c_1, c_2) = e_c(L^{-1} c_1 - L^{-1} c_2),$$

where L^{-1} denotes the inverse of L, which is also a continuous self-adjoint positive linear operator. By the above expression, the constrained minimization problem (4) is simplified to a unconstrained one with the objective function

$$f(c_1, c_2) = F(c_1, c_2; \Phi(c_1, c_2)).$$

The Fréchet partial derivatives $\frac{\partial f}{\partial c_i} v_i$ of f with respect to c_i along direction v_i can be found as

$$\frac{\partial f}{\partial c_i} v_i = \int_{D_s} v_i \left[Z_i e_c \Phi + k_B T \ln \frac{c_i}{I_s} - k_B T \ln(1 - \nu(c_1 + c_2)) \right] d\mathbf{r}, \quad i = 1, 2,$$

where $Z_1 = 1$, and $Z_2 = -1$. From the stationary equations $\frac{\partial f}{\partial c_i} v_i = 0$ for $i = 1, 2$ it follows that the minimizer (C_+, C_-) must satisfy the following system of two equations:

$$Z_i e_c \Phi + k_B T \ln \frac{c_i}{I_s} - k_B T \ln(1-\nu(c_1+c_2)) = 0, \quad i=1,2.$$

We can further find the second Fréchet derivative $F''(c)$, and show that $F''(c)$ is a strictly positive linear operator, implying that the energy functional $F(c_1, c_2)$ has a unique minimizer.

Let $u = \frac{e_c}{k_B T}\Phi$. We rewrite the above equations as

$$c_1 = I_s[1-\nu(c_1+c_2)]e^{-u}, \quad c_2 = I_s[1-\nu(c_1+c_2)]e^{u}. \tag{6}$$

Solving the above system for c_1 and c_2, we derive the expressions of optimal concentrations C_+ and C_- as follows: Adding them together, we can get

$$c_1 + c_2 = \frac{2I_s \cosh(u)}{1+2I_s \nu \cosh(u)},$$

which is applied to (6) to yield the expressions of c_1 and c_2:

$$c_1 = \frac{I_s e^{-u}}{1+2I_s \nu \cosh(u)}, \quad c_2 = \frac{I_s e^{u}}{1+2I_s \nu \cosh(u)}.$$

Substituting the above expressions to (5) gives the SMPBE in the solvent region D_s:

$$-\varepsilon_s \Delta u(\mathbf{r}) + \frac{2e_c^2 I_s \sinh(u)}{\varepsilon_0 k_B T[1+2I_s \nu \cosh(u)]} = 0, \qquad \mathbf{r} \in D_s, \quad u(\mathbf{s}) = 0, \quad \mathbf{s} \in \partial D_s, \tag{7}$$

where the length is in meters (m), and I_s is given in moles per liter (mol/l).

Clearly, the dimensionless electrostatic potential induced from the atomic charges of the protein is defined by the Poisson equation:

$$-\varepsilon_p \Delta u(\mathbf{r}) = \frac{e_c^2}{\varepsilon_0 k_B T} \sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, \quad \mathbf{r} \in D_p. \tag{8}$$

In the case of a protein immersed in the ionic solvent, the domain Ω is split to a protein region D_p, a solvent region D_s, and the interface Γ between them such that

$$\Omega = D_p \cup D_s \cup \Gamma.$$

From the classic linear dielectric theory,[30] it is known that the two potentials from D_p and D_s are related by the interface conditions

$$u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \frac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \frac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, \qquad \mathbf{s} \in \Gamma. \tag{9}$$

Thus, a combination of (9) with (7) and (8) results in an interface problem for computing the dimensionless electrostatic potential u over the domain Ω as follows:

$$\begin{cases} -\varepsilon_p \Delta u(\mathbf{r}) = \dfrac{e_c^2}{\varepsilon_0 k_B T} \displaystyle\sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\ -\varepsilon_s \Delta u(\mathbf{r}) + \dfrac{2e_c^2 I_s \sinh(u)}{\varepsilon_0 k_B T[1+2I_s \nu \cosh(u)]} = 0, & \mathbf{r} \in D_s, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ u(\mathbf{s}) = 0, & \mathbf{s} \in \partial\Omega. \end{cases} \tag{10}$$

The above model is called the SMPBE as it can be reduced to the classic Poisson-Boltzmann equation with the ionic volume $v = 0$.

In biomolecular calculation, the length unit for biomolecular data is angstrom (Å). Thus, we rescale ε_0 and I_s from meters (m) to angstroms as follows:

$$\varepsilon_0 \, \mathrm{F/m} = \frac{\varepsilon_0}{10^{10}} \mathrm{F/Å}, \quad I_s \, \mathrm{mole/liter} = I_s 10^3 N_A/m^3 = 10^{-27} N_A I_s/\mathrm{Å}^3, \tag{11}$$

where F denotes Farad (the unit of capacitance in SI), and we have used the unit converters: 1 m = 10^{10} Å, 1 liter = $10^3/m^3$, and 1 mole = N_A (Avogadro number), which estimates the number of ions as $N_A = 6.02214129 \times 10^{23}$.

Applying (11) to (10) and using $v = \Lambda^3$ (i.e., a volume of the cube with side length Λ), we obtain the SMPBE model that is used in our web server for computing biomolecular electrostatics.

Keywords: size modified Poisson-Boltzmann equation-interactive visualization · finite element method · biomolecular electrostatics · 3Dmol.js

How to cite this article: Y. Xie, J. Ying, D. Xie. *J. Comput. Chem.* **2017**, *38*, 541–552. DOI: 10.1002/jcc.24703

[1] N. A. Baker, *Rev. Comput. Chem.* **2005**, *21*, 349.
[2] B. Honig, A. Nicholls, *Science* **1995**, *268*, 1144.
[3] S. Unni, Y. Huang, R. M. Hanson, M. Tobias, S. Krishnan, W. W. Li, J. E. Nielsen, N. A. Baker, *J. Comput. Chem.* **2011**, *32*, 1488.
[4] N. Smith, S. Witham, S. Sarkar, J. Zhang, L. Li, C. Li, E. Alexov, *Bioinformatics* **2012**, *28*, 1655.
[5] R. Luo, L. David, M. K. Gilson, *J. Comput. Chem.* **2002**, *23*, 1244.
[6] C. Wang, J. Wang, Q. Cai, Z. Li, H. K. Zhao, R. Luo, *Comput. Theor. Chem.* **2013**, *1024*, 34.
[7] M. E. Davis, J. D. Madura, B. A. Luty, J. A. McCammon, *Comput. Phys. Commun.* **1991**, *62*, 187.
[8] S. Jo, M. Vargyas, J. Vasko-Szedlar, B. Roux, W. Im, *Nucleic Acids Res.* **2008**, *36*, W270.
[9] S. Jo, T. Kim, V. G. Iyer, W. Im, *J. Computat. Chem.* **2008**, *29*, 1859.
[10] V. Chu, Y. Bai, J. Lipfert, D. Herschlag, S. Doniach, *Biophys. J.* **2007**, *93*, 3202.
[11] I. Borukhov, D. Andelman, H. Orland, *Phys. Rev. Lett.* **1997**, *79*, 435.
[12] B. Li, *Nonlinearity* **2009**, *22*, 811.
[13] B. Eisenberg, In *Advances in Chemical Physics*, S. A. Rice, Ed.; Wiley: New York, **2011**; pp. 77–223.
[14] P. Koehl, M. Delarue, *J. Chem. Phys.* **2010**, *132*, 064101.
[15] P. Koehl, H. Orland, M. Delarue, *Phys. Rev. Lett.* **2009**, *102*, 087801.
[16] P. Koehl, F. Poitevin, H. Orland, M. Delarue, *J. Theor. Comput. Chem.* **2014**, *13*, 1440001.

[17] M. Z. Bazant, B. D. Storey, A. A. Kornyshev, *Phys. Rev. Lett.* **2011**, *106*, 046102.
[18] G. Tresset, *Phys. Rev. E* **2008**, *78*, 061506.
[19] C. D. Santangelo, *Phys. Rev. E* **2006**, *73*, 041512.
[20] D. Xie, J. L. Liu, B. Eisenberg, *Phys. Rev. E* **2016**, *94*, 012114.
[21] J. Li, D. Xie, *Int. J. Numer. Anal. Model.* **2015**, *12*, 286.
[22] J. Ying, D. Xie, A hybrid solver of size modified Poisson-Boltzmann equation by domain decomposition, finite element, and finite difference, **2016**, arXiv:1610.06173 [math.NA].
[23] D. Xie, *J. Comput. Phys.* **2014**, *275*, 294.
[24] J. Ying, D. Xie, *J. Comput. Phys.* **2015**, *298*, 636.
[25] A. Logg, K.-A. Mardal, G. N. Wells, Eds. *Automated Solution of Differential Equations by the Finite Element Method, Vol. 84 of Lecture Notes in Computational Science and Engineering;* Berlin Heidelberg, Springer Verlag, **2012**.
[26] Y. Jiang, Y. Xie, J. Ying, D. Xie, Z. Yu, *Mol. Based Math. Biol.* **2015**, *3*, 179.
[27] N. Rego, D. Koes, *Bioinformatics* **2015**, *31*, 1322.
[28] S. Brenner, L. Scott, *The Mathematical Theory of Finite Element Methods;* Springer-Verlag: New York, **2008**, 3rd ed.
[29] D. Xie, J. Li, *Nonlinear Anal. Real World Appl.* **2015**, *21*, 185.
[30] D. Griffiths, *Introduction to Electrodynamics*, 3rd ed.; Prentice Hall: New Jersey, **1999**.
[31] Z. Yu, M. Holst, Y. Cheng, J. McCammon, *J. Mol. Graph. Model.* **2008**, *26*, 1370.
[32] D. Xu, Y. Zhang, *PloS One* **2009**, *4*, e8140.
[33] C. Bertonati, B. Honig, E. Alexov, *Biophys. J.* **2007**, *92*, 1891.
[34] C. García-García, D. Draper, *J. Mol. Biol.* **2003**, *331*, 75.
[35] D. Sitkoff, K. A. Sharp, B. Honig, *J. Phys. Chem.* **1994**, *98*, 1978.
[36] G. S. Manning, *Q. Rev. Biophys.* **1978**, *11*, 179.
[37] K. J. Breslauer, D. P. Remeta, W. Y. Chou, R. Ferrante, J. Curry, D. Zaunczkowski, J. G. Snyder, L. A. Marky, *Proc. Natl. Acad. Sci. USA* **1987**, *84*, 8922.

Received: 26 October 2016
Revised: 21 November 2016
Accepted: 27 November 2016
Published online on 4 January 2017